REMARKS

Applicant amended independent claim 22 to correct a typographical error, but otherwise applicant did not recite new features in independent claim 22. Applicant also amended independent claims 1 and 17 to make them consistent with the language of independent claim 22 so as to clarify the inherent feature of independent claims 1 and 17 that the value based upon-which a branch operation is performed is a specified value in the branch instruction. Applicant further amended claim 17 to correct a typographical error. Claims 1-4, 6-14, 17-26 are pending. Claims 1, 17, and 22 are independent.

The examiner rejected claims 1, 17, 19-22 under 35 U.S.C. 102(a) and (b) as being anticipated by U.S. Patent No. 5,724,563 to Hasegawa. The examiner also rejected claims 1-4, 6-9, 11-13 under 35 U.S.C. §103(a) as being unpatentable over Hasegawa in view of U.S. Patent No. 5,394,530 to Kitta.

Additionally, the examiner rejected claim 10 under 35 U.S.C. §103(a) as being unpatentable over Hasegawa in view of Kitta, and further in view of U.S. Patent No. 5,274,770 to Khim Yeoh et al., rejected claim 14 under 35 U.S.C. §103(a) as being unpatentable over Hasegawa in view of Kitta, and further in view of U.S. Patent No. 4,742,451 to Brucker et al.. The examiner further rejected claim 18 under 35 U.S.C. §103(a) as being unpatentable over Hasegawa in view Brucker, rejected claims 24-26 under 35 U.S.C. §103(a) as being unpatentable over Hasegawa in view of U.S. Patent No. 5,923,872 to Chrysos, and rejected claim 23 under 35 U.S.C. §103(a) as being unpatentable over Hasegawa in view U.S. Patent No. 5,463,746 to Brodnax et al.

With respect to independent claim 22, the examiner contends that:

> **13. As to claim 22, Hasegawa also taught processor comprising:**
> **a) decode logic for decoding instructions (see decoding section in col.6, lines 43-50), the decode logic including logic to execute a branch instruction in execution of an instruction stream in the processor, with a branch operation based on any specified value in the branch instruction being true or false (se the conditional comparison in col.11, lines 20-54) ... (emphasis added, paragraph 13 at page 4 of the Office Action)**

Applicant respectfully disagrees.

Applicant's independent claim 22 recites "logic to: execute a branch instruction in execution of an instruction stream in the processor, with a branch operation based on a *specified value in the branch instruction* being true or false and including a token that specifies the number of instructions in the instruction stream to execute before performing the branch operation" (emphasis added). Thus, applicant's branch instruction causes an evaluation of whether a value, specified in the branch instruction, is true or false. Based on that evaluation, the branch instruction also causes a branching operation to be performed. In other words, applicant's branch instruction causes both an evaluation operation and a branching operation to be performed.

Hasegawa, on the other hand, discloses a pipeline processor that can execute predictive branch instructions (Abstract). As shown in FIG. 9, and as described in column 11, predictive branching is performed based on the execution result 109 produced by executing an instruction on execution section 11 of Hasegawa's pipeline processor. Specifically, the execution of an instruction on execution section 11 (the executed instruction is not the predictive branch instruction) causes a number of flags to be set in accordance with the result outcome produced by the execution of the instruction (col. 11, lines 25-32). These flags include the Zero flag Z, the negative flag N, the Carryover flag C, and the overflow flag V.

Hasegawa further explains that subsequently, after the execution result 109 is produced, the opcode in the region 21 of a predictive branch instruction (see Hasegawa's FIG. 2) is input from the instruction decoding section 3 to the branch condition judging section 62 (col. 11, lines 33-35). For example, if the opcode received from the predictive branching instruction is "100", branching will occur if the Z flag has been set as a result of the execution of a preceding instruction (see Table 1 at col. 11). As can also be seen from Hasegawa's FIG. 2, Hasegawa's predictive branch instruction does not include a true/false value that is used to determine if branching is to be performed. Thus, Hasegawa's predictive branch instruction does not cause an evaluation of whether a value is true or false, and certainly Hasegawa's predictive branch instruction does not itself specify what value that is to be evaluated. Rather, Hasegawa's predictive branch instruction causes a branching operation to be performed based on values previously evaluated through the execution of an earlier instruction. Accordingly, Hasegawa does not disclose or suggest "logic to: execute a branch instruction in execution of an instruction

stream in the processor, with a branch operation based on a specified value in the branch instruction being true or false and including a token that specifies the number of instructions in the instruction stream to execute before performing the branch operation," as required by applicant's independent claim 22.

Kitta describes improved techniques for predicting a branch target address using a branch history table (Abstract). While Kitta generally discusses branching instructions, nowhere does Kitta describe the basis upon which branch instructions are performed. Kitta, therefore, does not describe that branching is performed based on a specified value in the branch instruction being true or false. Thus, Kitta does not disclose or suggest "logic to: execute a branch instruction in execution of an instruction stream in the processor, with a branch operation based on a specified value in the branch instruction being true or false and including a token that specifies the number of instructions in the instruction stream to execute before performing the branch operation," as required by applicant's independent claim 22.

Accordingly, since neither Hasegawa not Kitta discloses or suggests, alone or in combination, the feature of "logic to: execute a branch instruction in execution of an instruction stream in the processor, with a branch operation based on a specified value in the branch instruction being true or false and including a token that specifies the number of instructions in the instruction stream to execute before performing the branch operation," applicant's independent claim 22 is therefore patentable over the cited art.

Claims 23 and 26 depend from applicant's independent claim 22 and are therefore patentable for at least the same reasons as independent claim 22.

Applicant's independent claims 1 and 17 recite "executing a branch instruction in execution of an instruction stream with a branch based on a specified value in the branch instruction being true or false," or similar language. For reasons similar to those provided with respect to applicant's independent claim 22, at least this feature is not disclosed by the cited art. Accordingly, applicant's independent claims 1 and 17 are patentable over the cited art.

Claims 2-4, 6-14, and 24 depend from independent claim 1 and are therefore patentable for at least the same reasons as independent claim 1. Claims 18-21 and 25 depend from independent claim 17 and are therefore patentable for at least the same reasons as independent claim 17.

Furthermore, as noted above, the examiner rejected claim 24-26 under 35 U.S.C. §103(a) as being unpatentable over Hasegawa in view of Chrysos. Specifically, the examiner admits that "[a]s to claims 24-26 ... Hasegawa did not specifically teach the hardware based multi threaded as claimed." (paragraph 35 on page 11 of the Office Action). The examiner, however, relies on Chrysos for disclosing the above feature, and argues that "Chrysos discloses a system including hardware contexts for simultaneously multithreaded execution (see col. 12, lines 15-18)" (paragraph 35, page 11 of the Office Action).

Applicant's claim 24 recites "[t]he method of claim 1, wherein the processor is a hardware-based multithreaded processor having multiple engines to process multiple threads and the branch instruction is part of an instruction set for the multiple engines; and executing comprises: executing the branch on one of the multiple engines." Thus, applicant's processor is a multithreaded processor that has multiple engines that can each process multiple threads. The multithreaded processor coordinates and oversees the operation of its thread-processing engines.

In contrast, Chrysos describes an apparatus for sampling values of operands of instructions in a processor pipeline of a system that has a plurality of processing stages (Abstract). Chrysos discloses that the computer system 100, shown in FIG. 1, may include one or more processors 110 (col. 7, line 66 to col. 8, line 1), and that some of those processors may have multiple logical registers sets, thereby enabling processing of simultaneously executing threads (col. 12, lines 15-18). But nowhere does Chrysos disclose a processor structure that includes a main multithreaded processor that has multiple engines to process multiple threads. Thus, Chrysos does not disclose or suggest a processor that "is a hardware-based multithreaded processor having multiple engines to process multiple threads and the branch instruction is part of an instruction set for the multiple engines; and executing comprises: executing the branch on one of the multiple engines," as required by applicant's claim 24.

Since neither Hasegawa nor Chrysos discloses or suggests, alone or in combination, at least the feature of "a hardware-based multithreaded processor having multiple engines to process multiple threads and the branch instruction is part of an instruction set for the multiple engines; and executing comprises: executing the branch on one of the multiple engines," applicant's claim 24 is therefore patentable over the cited art.

Claims 25 and 26 recite "a hardware-based multithreaded processor having multiple engines to process multiple threads and the branch instruction is part of an instruction set for the multiple engines; and executing and deferring occurs on one of the multiple engines," or similar language. For reasons similar to those provided with respect to claim 24, at least this feature is not disclosed by the cited art. Accordingly, claims 25 and 26 are patentable over the cited art.

It is believed that all the rejections and/or objections raised by the examiner have been addressed.

All of the dependent claims are patentable for at least the reasons for which the claims on which they depend are patentable.

Canceled claims, if any, have been canceled without prejudice or disclaimer.

Any circumstance in which the applicant has (a) addressed certain comments of the examiner does not mean that the applicant concedes other comments of the examiner, (b) made arguments for the patentability of some claims does not mean that there are not other good reasons for patentability of those claims and other claims, or (c) amended or canceled a claim does not mean that the applicant concedes any of the examiner's positions with respect to that claim or other claims.

No fees are believed due. Please apply any other charges to deposit account 06-1050, referencing attorney docket 10559-311US1.

Respectfully submitted,

Date: Dec. 27, 2005

Ido Rabinovitch
Attorney for Intel Corporation
Reg. No. L0080

Fish & Richardson P.C.
Telephone: (617) 542-5070
Facsimile: (617) 542-8906

21235135.doc